Second-Order Algebra and Generalised Polynomial Functors

Marcelo Fiore

Computer Laboratory
University of Cambridge

LI 2012
Algebra and Computation
27.II.2012

I

Second-Order Algebra

The algebraic theory of languages
with variable binding

Second-Order Equational Presentations ∼ Examples ∼

1. **Indefinite summation**

Sorts

Expressions	ε	:	$*$

Operators

Addition	$+$	:	$\varepsilon\,,\varepsilon \longrightarrow \varepsilon$
Summation	$\sum$	:	$(\varepsilon)\varepsilon \longrightarrow \varepsilon$

Second-Order Equational Presentations ∼ Examples ∼

1. **Indefinite summation**

Sorts

Expressions $\quad \varepsilon \; : \; *$

Operators

Addition $\quad + \; : \; \varepsilon\,,\varepsilon \longrightarrow \varepsilon$

Summation $\quad \sum \; : \; (\varepsilon)\varepsilon \longrightarrow \varepsilon$

Axioms

$$\sum\Big(i.\sum(j.\,\mathsf{E}[i,j])\Big) \equiv \sum\Big(j.\sum(i.\,\mathsf{E}[i,j])\Big)$$

$$\sum(i.\,\mathsf{E}[i]) + \sum(j.\,\mathsf{F}[j]) \equiv \sum\Big(k.\,\mathsf{E}[k] + \mathsf{F}[k]\Big)$$

Second-Order Equational Presentations ∼ Examples ∼

1. **Indefinite summation**

Sorts

Expressions $\qquad \varepsilon \quad : \quad *$

Operators

Addition $\qquad + \quad : \quad \varepsilon\,,\varepsilon \longrightarrow \varepsilon$

Summation $\qquad \sum \quad : \quad (\varepsilon)\varepsilon \longrightarrow \varepsilon$

Axioms

$$\mathsf{E} : [\varepsilon,\varepsilon]\varepsilon$$

$$\vdash \sum\big(i.\sum(j.\,\mathsf{E}[i,j])\big) \equiv \sum\big(j.\sum(i.\,\mathsf{E}[i,j])\big) : \varepsilon$$

$$\mathsf{E} : [\varepsilon]\varepsilon\,,\mathsf{F} : [\varepsilon]\varepsilon$$

$$\vdash \sum(i.\,\mathsf{E}[i]) + \sum(j.\,\mathsf{F}[j]) \equiv \sum\big(k.\,\mathsf{E}[k] + \mathsf{F}[k]\big) : \varepsilon$$

2. **Definite summation**

Sorts

Expressions $\varepsilon \quad : \quad *$

Operators

One $1 \quad : \quad \varepsilon$

Addition $+ \quad : \quad \varepsilon\,,\varepsilon \longrightarrow \varepsilon$

Summation $\sum \quad : \quad \varepsilon\,,\varepsilon\,,(\varepsilon)\varepsilon \longrightarrow \varepsilon$

Axioms

$$\mathsf{A},\mathsf{B} : \varepsilon\,,\mathsf{E} : [\varepsilon]\varepsilon$$

$$\vdash \sum\left(\mathsf{A}\,,\mathsf{B}+1\,,i.\,\mathsf{E}[i]\right)$$

$$\equiv \sum\left(\mathsf{A}\,,\mathsf{B}\,,i.\,\mathsf{E}[i]\right) + \mathsf{E}\left[\mathsf{B}+1\right] : \varepsilon$$

3. **Classical first-order logic**

Sorts

Individuals	ι	:	$*$
Formulas	φ	:	$*$

Operators

Connectives	$\bot, \top$	:	φ
	$\vee, \wedge$	:	$\varphi\,,\varphi \to \varphi$
	$\neg$	:	$\varphi \to \varphi$
Functions	$f_i^{(m)}$	:	$\underbrace{\iota, \ldots, \iota}_{m} \to \iota$
Predicates	$P_j^{(n)}$	:	$\underbrace{\iota, \ldots, \iota}_{n} \to \varphi$
Quantifiers	$\forall$	:	$(\iota)\varphi \to \varphi$
	$\exists$	:	$(\iota)\varphi \to \varphi$

Axioms

Boolean algebra axioms for $(\bot, \vee, \top, \wedge, \neg)$

$$\mathsf{P} : [\iota]\varphi\ ,\ \ \mathsf{X} : \iota$$

$$\vdash\ \forall\big(x.\,\mathsf{P}[x]\big) \equiv \forall\big(x.\,\mathsf{P}[x]\big) \wedge \mathsf{P}[\mathsf{X}]\ : \varphi$$

$$\mathsf{P} : [\iota]\varphi\ ,\ \ \mathsf{Q} : \varphi$$

$$\vdash\ \forall\big(x.\,\mathsf{P}[x] \vee \mathsf{Q}\big) \equiv \forall\big(x.\,\mathsf{P}[x]\big) \vee \mathsf{Q}\ : \varphi$$

$$\mathsf{P} : [\iota]\varphi$$

$$\vdash\ \neg\big(\exists(x.\,\mathsf{P}[x])\big) \equiv \forall\big(x.\,\neg(\mathsf{P}[x])\big) : \varphi$$

Theory axioms $\vdash \phi_k \equiv \top : \varphi$

4. **Untyped lambda calculus**

Sorts

Lambda terms $\Lambda \ : \ *$

Operators

Application $@ \ : \ \Lambda\,,\Lambda \to \Lambda$

Abstraction $\lambda \ : \ (\Lambda)\Lambda \to \Lambda$

Axioms

(β) $\mathsf{M} : [\Lambda]\Lambda\,,\mathsf{N} : \Lambda$

$$\vdash \lambda\big(x.\,\mathsf{M}[x]\big) \mathbin{@} \mathsf{N} \equiv \mathsf{M}[\mathsf{N}] : \Lambda$$

(η) $\mathsf{F} : \Lambda$

$$\vdash \lambda\big(x.\,\mathsf{F} \mathbin{@} x\big) \equiv \mathsf{F} : \Lambda$$

5. **Simply-typed lambda calculus**

Sorts

Basic types	β	:	$*$
Arrow types	$\Rightarrow$	:	$*, * \longrightarrow *$

Operators

Application	$@^{S,T}$	:	$S \Rightarrow T, S \rightarrow T$
Abstraction	$\lambda^{S,T}$	:	$(S)T \rightarrow S \Rightarrow T$

Axioms

$$(\beta^{S,T})\ \ M : [S]T, N : S \vdash \lambda^{S,T}(x.\, M[x])\ @^{S,T} N \equiv M[N] : T$$

$$(\eta^{S,T})\ \ F : S \Rightarrow T \vdash \lambda^{S,T}(x.\, F\ @^{S,T} x) \equiv F : S \Rightarrow T$$

Second-Order Algebraic Syntax

Operators

$$o : (\vec{\sigma_1})\tau_1, \ldots, (\vec{\sigma_n})\tau_n \to \tau$$

o is an operator of sort τ taking n arguments each of which binds, for $\vec{\sigma_i} = \sigma_{i,1}, \ldots, \sigma_{i,n_i}$, n_i variables of sorts $\sigma_{i,1}, \ldots, \sigma_{i,n_i}$ in a term of sort τ_i.

Typing contexts

$$M_1 : [\vec{\sigma_1}]\tau_1, \ldots, M_k : [\vec{\sigma_k}]\tau_k \triangleright x_1 : \sigma'_1, \ldots, x_\ell : \sigma'_\ell$$

Typing contexts

$$M_1 : [\vec{\sigma_1}]\tau_1, \ldots, M_k : [\vec{\sigma_k}]\tau_k \rhd x_1 : \sigma'_1, \ldots, x_\ell : \sigma'_\ell$$

Terms

(Variables)

For $(x : \tau) \in \Gamma$,

$$\frac{}{\Theta \rhd \Gamma \vdash x : \tau}$$

Typing contexts

$$M_1 : [\vec{\sigma_1}]\tau_1, \ldots, M_k : [\vec{\sigma_k}]\tau_k \triangleright x_1 : \sigma'_1, \ldots, x_\ell : \sigma'_\ell$$

Terms

(Variables)

For $(x : \tau) \in \Gamma$,

$$\frac{}{\Theta \triangleright \Gamma \vdash x : \tau}$$

(Parameterised metavariables)

For $(M : [\tau_1, \ldots, \tau_n]\tau) \in \Theta$,

$$\frac{\Theta \triangleright \Gamma \vdash t_i : \tau_i \ \ (1 \leq i \leq n)}{\Theta \triangleright \Gamma \vdash M[t_1, \ldots, t_n] : \tau}$$

(Operators)

For $\mathsf{o} : (\vec{\sigma_1})\tau_1, \ldots, (\vec{\sigma_n})\tau_n \to \tau$,

$$\frac{\Theta \rhd \Gamma, \vec{x_i} : \vec{\sigma_i} \vdash t_i : \tau_i \; (1 \le i \le n)}{\Theta \rhd \Gamma \vdash \mathsf{o}\big(\vec{x_1}.\, t_1, \ldots, \vec{x_n}.\, t_n\big) : \tau}$$

where $\vec{x} : \vec{\sigma}$ stands for $x_1 : \sigma_1, \ldots, x_k : \sigma_k$.

An *equational presentation* is a set of axioms each of which is a pair of terms in context.

(Operators)

For $\mathsf{o} : (\vec{\sigma_1})\tau_1, \ldots, (\vec{\sigma_n})\tau_n \to \tau$,

$$\frac{\Theta \rhd \Gamma, \vec{x_i} : \vec{\sigma_i} \vdash t_i : \tau_i \ (1 \leq i \leq n)}{\Theta \rhd \Gamma \vdash \mathsf{o}\big(\vec{x_1}.\, t_1, \ldots, \vec{x_n}.\, t_n\big) : \tau}$$

where $\vec{x} : \vec{\sigma}$ stands for $x_1 : \sigma_1, \ldots, x_k : \sigma_k$.

Equational presentations

$$\mathcal{E} = \{\, \Theta_i \rhd_i \Gamma_i \vdash s_i \equiv t_i : \tau_i \,\}_{i \in I}$$

An *equational presentation* is a set of axioms each of which is a pair of terms in context.

Second-Order Algebraic Models

Algebras over abstract clones

Second-Order Algebraic Models

Algebras over abstract clones

- *Signature models*
- *Equational-presentation models*

Second-Order Algebraic Models

Algebras over abstract clones

- *Signature models* are *abstract clones* together with *compatible operator interpretations*.
- *Equational-presentation models* are signature models that *satisfy the axioms*.

S-sorted Abstract Clones

Monoids

$$V \to A \leftarrow A \bullet A \quad \text{in} \quad \left(\boldsymbol{\mathcal{S}et}^{\mathbf{F}\downarrow S}\right)^S$$

for $\mathbf{F}$ a skeleton of finite sets with respect to the *substitution monoidal structure*

- $V_\tau = \boldsymbol{y}(\tau)$
- $(X \bullet Y)_\tau = \int^{\vec{\sigma} \in \mathbf{F}\downarrow S} X_\tau(\vec{\sigma}) \times \prod_{\sigma_i \in \vec{\sigma}} Y_{\sigma_i}$

NB. The unit and multiplication operations of abstract clones respectively provide interpretations for variables and metavariables.

Signature Algebras

Algebras

$$\Sigma A \to A \ \text{ in } \ \left(\mathcal{S}\boldsymbol{et}^{\mathbf{F}\downarrow S}\right)^{S}$$

for

$$\Sigma(X)_\tau = \coprod_{o:(\vec{\sigma_1})\tau_1,\ldots,(\vec{\sigma_n})\tau_n \to \tau} \prod_{1 \le i \le n} X^{\boldsymbol{y}(\vec{\sigma_i})}$$

Signature Models

- Monoid structure:

$$V \xrightarrow{e} A \xleftarrow{m} A \bullet A$$

- Algebra structure:

$$\Sigma A \xrightarrow{\xi} A$$

subject to the compatibility condition:

$$\begin{array}{ccccc}
\Sigma(A) \bullet A & \longrightarrow & \Sigma(A \bullet A) & \xrightarrow{\Sigma m} & \Sigma A \\
{\scriptstyle \xi \bullet A}\downarrow & & & & \downarrow{\scriptstyle \xi} \\
A \bullet A & & \xrightarrow[m]{\hspace{4em}} & & A
\end{array}$$

Monadic Signature Models

$$\mathrm{Mod}(\Sigma) \underset{\longrightarrow}{\overset{\longleftarrow}{\dashv}} \left(\boldsymbol{\mathcal{S}et}^{\mathbf{F}\downarrow S}\right)^{S} \quad \circlearrowleft \mathcal{M}_\Sigma$$

1. $\mathcal{M}_\Sigma(X) \cong V + X \bullet \mathcal{M}_\Sigma(X) + \Sigma(\mathcal{M}_\Sigma X)$
2. Free constructions describe syntax with variable binding and parameterised metavariables.

   Terms of sort τ in context

   $$\mathsf{M}_1 : [\vec{\sigma_1}]\tau_1, \ldots, \mathsf{M}_k : [\vec{\sigma_k}]\tau_k \rhd x_1 : \sigma'_1, \ldots, x_\ell : \sigma'_\ell$$

   are in bijective correspondence with Kleisli maps

   $$\boldsymbol{y}(\sigma'_1, \ldots, \sigma'_\ell)_{@\tau} \to \mathcal{M}_\Sigma\left(\coprod_{1\leq i\leq k} \boldsymbol{y}(\vec{\sigma_i})_{@\tau_i}\right)$$

3. The monoid multiplication

$$\mathcal{M}_\Sigma(\mathsf{X}) \bullet \mathcal{M}_\Sigma(\mathsf{X}) \to \mathcal{M}_\Sigma(\mathsf{X})$$

provides a definition of capture-avoiding simultaneous substitution by structural recursion.

4. $\mathcal{M}_\Sigma$ is a strong monad.

The strength

$$\mathcal{M}_\Sigma(\mathsf{X}) \times \prod_{\sigma \in S} \mathsf{Y}_\sigma{}^{\mathsf{X}_\sigma} \to \mathcal{M}_\Sigma(\mathsf{Y})$$

provides a definition of metavariable substitution by structural recursion.

Equational-Presentation Models

- The interpretation

$$[\![\Theta \rhd \Gamma \vdash t : \tau]\!]_{A}$$

of a term

$$\Theta \rhd \Gamma \vdash t : \tau$$

in a signature model A, where

$$\Theta = \mathsf{M}_1 : [\vec{\sigma_1}]\tau_1, \ldots, \mathsf{M}_k : [\vec{\sigma_k}]\tau_k$$

and

$$\Gamma = x_1 : \sigma'_1, \ldots, \sigma'_\ell \quad ,$$

is a map

$$\prod_{1 \leq i \leq k} A_{\tau_i}{}^{y(\vec{\sigma_i})} \to A_{\tau}{}^{y(\vec{\sigma'})} \text{ in } \mathcal{S}et^{\mathbf{F}\downarrow S} \quad .$$

- $\mathrm{Mod}(\Sigma, \mathcal{E})$ is the full subcategory of $\mathrm{Mod}(\Sigma)$ consisting of all those signature models A for which

$$[\![\Theta \rhd \Gamma \vdash s : \tau]\!]_A = [\![\Theta \rhd \Gamma \vdash t : \tau]\!]_A$$

for all axioms $\Theta \rhd \Gamma \vdash s \equiv t : \tau$ in $\mathcal{E}$.

- $\mathrm{Mod}(\Sigma, \mathcal{E})$ is the full subcategory of $\mathrm{Mod}(\Sigma)$ consisting of all those signature models A for which

$$[\![\Theta \rhd \Gamma \vdash s : \tau]\!]_A = [\![\Theta \rhd \Gamma \vdash t : \tau]\!]_A$$

  for all axioms $\Theta \rhd \Gamma \vdash s \equiv t : \tau$ in $\mathcal{E}$.

- $\mathrm{Mod}(\Sigma, \mathcal{E})$ is monadic over $\left(\mathcal{S}et^{\mathbf{F}\downarrow S}\right)^S$ with inductively constructed free models. The induced monad is finitary and preserves epimorphisms.

- $\mathrm{Mod}(\Sigma, \mathcal{E})$ is complete and cocomplete.

Second-Order Equational Logic

The second-order nature of the syntax requires a *two-level* substitution calculus.

Second-Order Equational Logic

The second-order nature of the syntax requires a *two-level* substitution calculus.

Substitution calculus

- **Capture-avoiding simultaneous substitution** of terms for variables.

  Maps

  $$\Theta \rhd x_1 : \sigma_1, \ldots, x_n : \sigma_n \vdash t : \tau$$

  and

  $$\Theta \rhd \Gamma \vdash t_i : \sigma_i \quad (1 \leq i \leq n)$$

  to

  $$\Theta \rhd \Gamma \vdash t[{}^{t_i}\!/\!_{x_i}]_{1 \leq i \leq n} : \tau$$

- **Metasubstitution** of abstracted terms for metavariables.

  Maps

  $$\mathsf{M}_1 : [\vec{\sigma_1}]\tau_1, \ldots, \mathsf{M}_k : [\vec{\sigma_k}]\tau_k \rhd \Gamma \vdash t : \tau$$

  and

  $$\Theta \rhd \Gamma, \vec{x_i} : \vec{\sigma_i} \vdash t_i : \tau_i \quad (1 \leq i \leq k)$$

  to

  $$\Theta \rhd \Gamma \vdash t\{\mathsf{M}_i := (\vec{x_i})t_i\}_{1 \leq i \leq k} : \tau$$

- **Metasubstitution** of abstracted terms for metavariables.

Maps

$$M_1 : [\vec{\sigma_1}]\tau_1, \ldots, M_k : [\vec{\sigma_k}]\tau_k \rhd \Gamma \vdash t : \tau$$

and

$$\Theta \rhd \Gamma, \vec{x_i} : \vec{\sigma_i} \vdash t_i : \tau_i \quad (1 \le i \le k)$$

to

$$\Theta \rhd \Gamma \vdash t\{M_i := (\vec{x_i})t_i\}_{1 \le i \le k} : \tau$$

Definition:

- $x\{M_i := (\vec{x_i})t_i\}_{1 \le i \le k} = x$
- $\big(M_\ell[s_1, \ldots, s_m]\big)\{M_i := (\vec{x_i})t_i\}_{1 \le i \le k}$
  $= \; t_\ell[s'_j/x_{i,j}]_{1 \le j \le m}$

  where $s'_j = s_j\{M_i := (\vec{x_i})t_i\}_{1 \le i \le k}$
- $\big(o(\ldots, (\vec{x})s, \ldots)\big)\{M_i := (\vec{x_i})t_i\}_{1 \le i \le k}$

  $= \; o\big(\ldots, (\vec{x})s\{M_i := (\vec{x_i})t_i\}_{1 \le i \le k}, \ldots\big)$

Deductive system

(Extended metasubstitution)

$$\frac{M_1 : [\vec{\sigma_1}]\tau_1, \ldots, M_k : [\vec{\sigma_k}]\tau_k \rhd \Gamma \vdash s \equiv t : \tau \qquad \Theta \rhd \Delta, \vec{x_i} : \vec{\sigma_i} \vdash s_i \equiv t_i : \tau_i \quad (1 \leq i \leq k)}{\Theta \rhd \Gamma, \Delta \vdash s\{M_i := (\vec{x_i})s_i\}_{1 \leq i \leq k} \equiv t\{M_i := (\vec{x_i})t_i\}_{1 \leq i \leq k} : \tau}$$

Deductive system

(Extended metasubstitution)

$$\frac{\mathsf{M}_1 : [\vec{\sigma_1}]\tau_1, \ldots, \mathsf{M}_k : [\vec{\sigma_k}]\tau_k \rhd \Gamma \vdash s \equiv t : \tau \qquad \Theta \rhd \Delta, \vec{x_i} : \vec{\sigma_i} \vdash s_i \equiv t_i : \tau_i \quad (1 \leq i \leq k)}{\Theta \rhd \Gamma, \Delta \vdash s\{\mathsf{M}_i := (\vec{x_i})s_i\}_{1 \leq i \leq k} \equiv t\{\mathsf{M}_i := (\vec{x_i})t_i\}_{1 \leq i \leq k} : \tau}$$

We have:

- Conservativity over equational logic.
- Semantic completeness of second-order derivability.
- Derivability completeness of (bidirectional) second-order term rewriting.

Second-Order Theory of Equality

- *Mono-sorted* terms

$$M_1 : [m_1], \ldots, M_k : [m_k] \rhd x_1, \ldots, x_n \vdash s$$

where

$$\begin{array}{rcll} s & ::= & x_j & (1 \leq j \leq n) \\ & | & M_i[s_1, \ldots, s_{m_i}] & (1 \leq i \leq k) \end{array}$$

under the metasubstitution mechanism.

Second-Order Theory of Equality

- *Mono-sorted* terms

$$\mathsf{M}_1 : [m_1], \ldots, \mathsf{M}_k : [m_k] \rhd x_1, \ldots, x_n \vdash s$$

where

$$\begin{array}{rcll} s & ::= & x_j & (1 \leq j \leq n) \\ & \mid & \mathsf{M}_i[s_1, \ldots, s_{m_i}] & (1 \leq i \leq k) \end{array}$$

under the metasubstitution mechanism.

- The category $\mathbf{M}$ has set of objects $\mathbb{N}^*$ and morphisms

$$(m_1, \ldots, m_k) \to (n_1, \ldots, n_\ell)$$

given by tuples

$$\big\langle\, \mathsf{M}_1 : [m_1], \ldots, \mathsf{M}_k : [m_k] \rhd x_1, \ldots, x_{n_i} \vdash s_i \,\big\rangle_{1 \leq i \leq \ell}$$

that compose by *metasubstitution*.

The Structure of Second-Order Equality

Universal property of $\mathbb{M}$.

The Structure of Second-Order Equality

Universal property of $\mathbb{M}$.

The category $\mathbb{M}$ is universally characterised as the free (strict) cartesian category on an exponentiable object, *viz.* $(\!|0|\!)$.

The Structure of Second-Order Equality

Universal property of $\mathbb{M}$.

The category $\mathbb{M}$ is universally characterised as the free (strict) cartesian category on an exponentiable object, *viz.* (0).

- Products:

$$(m_1, \ldots, m_k) = (m_1) \times \cdots \times (m_k)$$

- Exponentiability:

$$(m) = (0)^m \Rightarrow (0)$$

Second-Order Algebraic Theories

- A (mono-sorted) *second-order algebraic theory* consists of a small cartesian category $\mathbb{T}$ and a strict cartesian identity-on-objects functor $\mathbf{M} \longrightarrow \mathbb{T}$ that preserves the exponentiable object (0).

Second-Order Algebraic Theories

- A (mono-sorted) *second-order algebraic theory* consists of a small cartesian category $\mathbb{T}$ and a strict cartesian identity-on-objects functor $\mathbf{M} \to \mathbb{T}$ that preserves the exponentiable object (0).

- The category $\mathcal{M}od(\mathbb{T})$ of (set-theoretic) *functorial models* of a second-order algebraic theory $\mathbb{T}$ is the category of cartesian functors $\mathbb{T} \to \mathcal{S}et$ and natural transformations between them.

Algebraic Translations

For second-order algebraic theories $\mathbf{M} \to \mathbb{T}$ and $\mathbf{M} \to \mathbb{T}'$, a second-order *algebraic translation* is a functor $\mathbb{T} \to \mathbb{T}'$ such that

$$\begin{array}{ccc} & \mathbf{M} & \\ \swarrow & & \searrow \\ \mathbb{T} & \longrightarrow & \mathbb{T}' \end{array}$$

Algebraic Translations

For second-order algebraic theories $\mathbf{M} \to \mathbb{T}$ and $\mathbf{M} \to \mathbb{T}'$, a second-order *algebraic translation* is a functor $\mathbb{T} \to \mathbb{T}'$ such that

$$\begin{array}{ccc} & \mathbf{M} & \\ \swarrow & & \searrow \\ \mathbb{T} & \longrightarrow & \mathbb{T}' \end{array}$$

Algebraic Functors

Every second-order algebraic translation $\mathrm{F} : \mathbb{T} \to \mathbb{T}'$ contravariantly induces an *algebraic functor* $\mathrm{F}^* : \mathcal{M}od(\mathbb{T}') \to \mathcal{M}od(\mathbb{T})$.

- Algebraic functors have left adjoints.

Theories *vs.* Presentations

Classifying categories

— the theory of a presentation

For every second-order equational presentation $\mathcal{E}$, we construct a second-order algebraic theory $\mathbf{M}(\mathcal{E})$.

Internal languages

— the presentation of a theory

For every second-order algebraic theory T, we construct a second-order equational presentation $\mathscr{E}(\mathsf{T})$.

- **Theory/presentation correspondence.**

  Every second-order algebraic theory T is isomorphic to the second-order algebraic theory of its associated equational presentation $\mathbf{M}(\mathscr{E}(\mathrm{T}))$.

- **Presentation/theory correspondence.**

  Every second-order equational presentation $\mathcal{E}$ is isomorphic, with respect to a notion of *syntactic translation*, to the second-order equational presentation of its associated algebraic theory $\mathscr{E}(\mathbf{M}(\mathcal{E}))$.

- The above two correspondences yield an equivalence of categories.

- **Universal-algebra/categorical-algebra correspondence.**

  For every second-order equational presentation $\mathcal{E}$, the category of algebraic models $\mathrm{Mod}(\mathcal{E})$ and the category of functorial models $\boldsymbol{\mathcal{M}od}(\mathrm{M}(\mathcal{E}))$ are equivalent.

- **Categorical-algebra/universal-algebra correspondence.**

  For every second-order algebraic theory T, the category of functorial models $\boldsymbol{\mathcal{M}od}(\mathsf{T})$ and the category of algebraic models $\mathrm{Mod}(\mathscr{E}(\mathsf{T}))$ are equivalent.

II

Generalised Polynomial Functors

Kan Extensions

Every

$$f : \mathbb{X} \to \mathbb{Y}$$

induces

$$\mathcal{P}\mathbb{X} \underset{f_!}{\overset{f_*}{\rightleftarrows}} \mathcal{P}\mathbb{Y}, \quad f_! \dashv f^* \dashv f_*$$

where

$$\mathcal{P}\mathbb{C} =^{\mathrm{def}} \boldsymbol{\mathcal{S}et}^{\mathbb{C}}$$

and

$$\begin{aligned} f_* P\, y &= \mathrm{Ran}_f P\, y = \int_{x \in \mathbb{X}} [\, \mathbb{Y}(y, fx) \Rightarrow Px \,] \\ f^* Q\, x &= Q(fx) \\ f_! P\, y &= \mathrm{Lan}_f P\, y = \int^{x \in \mathbb{X}} \mathbb{Y}(fx, y) \times Px \end{aligned}$$

Generalised Polynomial Functors

The class of

generalised polynomial functors

is the closure under natural isomorphism of the functors

$$\mathcal{P}\mathbb{A} \to \mathcal{P}\mathbb{B}$$

arising as composites

$$\mathcal{P}\mathbb{A} \xrightarrow{s^*} \mathcal{P}\mathbb{I} \xrightarrow{f_*} \mathcal{P}\mathbb{J} \xrightarrow{t_!} \mathcal{P}\mathbb{B}$$

from diagrams

$$\mathbb{A} \xleftarrow{s} \mathbb{I} \xrightarrow{f} \mathbb{J} \xrightarrow{t} \mathbb{B}$$

in $\mathcal{C}at$.

$$t_! f_* s^* A\, b = \int^{j \in \mathbb{J}} \mathbb{B}(tj, b) \times \int_{i \in \mathbb{I}} \left[\mathbb{J}(j, fi) \Rightarrow A(si) \right]$$

Examples:

- For every presheaf P, the product endofunctor $(-) \times P$ and the exponential endofunctor $(-)^P$ are generalised polynomial.

- Modulo the equivalence $\mathcal{P}(\mathbb{C})/P \simeq \mathcal{P}(\oint P)$, for $\oint P$ the category of elements of $P \in \mathcal{P}\mathbb{C}$, polynomial functors

$$\mathcal{P}(\mathbb{C})/A \xrightarrow{s^*} \mathcal{P}(\mathbb{C})/I \xrightarrow{\Pi_f} \mathcal{P}(\mathbb{C})/J \xrightarrow{\Sigma_t} \mathcal{P}(\mathbb{C})/B$$

for

$$A \xleftarrow{s} I \xrightarrow{f} J \xrightarrow{t} B \ \text{ in } \ \mathcal{P}(\mathbb{C})$$

are subsumed by generalised polynomial functors $\mathcal{P}(\oint A) \to \mathcal{P}(\oint B)$.

- Constant functors between presheaf categories are generalised polynomial.
- Every cocontinuous functor between presheaf categories is generalised polynomial.

Discrete Generalised Polynomial Functors

The class of *discrete* generalised polynomial functors is represented by diagrams of the form

$$\mathbb{A} \longleftarrow \coprod_{k \in K} L_k \cdot \mathbb{J}_k \xrightarrow{\coprod_{k \in K} \nabla_{L_k}} \coprod_{k \in K} \mathbb{J}_k \longrightarrow \mathbb{B}$$

where L_k is finite for all $k \in K$.

- Discrete generalised polynomial functors are finitary and preserve epimorphisms.

Examples:

- **Convolution monoidal closed structure**
  1. *Day's convolution tensor product* is [isomorphic to] a discrete generalised polynomial functor.
  2. *Exponentiation to a representable* with respect to the closed structure associated to the convolution monoidal structure is a discrete generalised polynomial functor.
- The substitution tensor product for planar operads is [isomorphic to] a discrete generalised polynomial functor.

- **Simply-typed lambda calculus syntax**

  Let S be the set of simple types.

  1. The rule

$$\frac{\Gamma \vdash t : \tau_1 \Rightarrow \tau_2 \qquad \Gamma \vdash t' : \tau_1}{\Gamma \vdash t \mathbin{@} t' : \tau_2}$$

  has associated the discrete generalised polynomial endofunctor represented by

$$\begin{array}{ccc}
2 \cdot \big((\mathbf{F}{\downarrow}S) \times S \times S\big) & \xrightarrow{\nabla_2} & (\mathbf{F}{\downarrow}S) \times S \times S \\
{\scriptstyle [\mathrm{id} \times \Rightarrow,\, \mathrm{id} \times \pi_1]} \downarrow & & \downarrow {\scriptstyle \mathrm{id} \times \pi_2} \\
(\mathbf{F}{\downarrow}S) \times S & & (\mathbf{F}{\downarrow}S) \times S
\end{array}$$

2. The rule

$$\frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda x.\, t : \tau_1 \Rightarrow \tau_2}$$

has associated the discrete generalised polynomial endofunctor represented by

$$\begin{array}{ccc}
(\mathbf{F}{\downarrow}S) \times S \times S & \xrightarrow{\mathrm{id}} & (\mathbf{F}{\downarrow}S) \times S \times S \\
{\scriptstyle +\times\mathrm{id}}\downarrow & & \downarrow{\scriptstyle \mathrm{id}\times\Rightarrow} \\
(\mathbf{F}{\downarrow}S) \times S & & (\mathbf{F}{\downarrow}S) \times S
\end{array}$$

- The class of discrete generalised polynomial functors is closed under
  - constants,
  - projections,
  - sums,
  - finite products,
  - composition, and
  - differentiation.

Differentiation

The *differential* of

$$\mathbb{A} \xleftarrow{\;s\;} L \cdot \mathbb{J} \xrightarrow{\;\nabla_L\;} \mathbb{J} \xrightarrow{\;t\;} \mathbb{B}$$

is the discrete polynomial

$$\begin{array}{ccc} \coprod_{(L_0,\ell_0)\in L'} L_0 \cdot \widetilde{\mathbb{J}} & \xrightarrow{\;\coprod_{(L_0,\ell_0)\in L'} \nabla_{L_0}\;} & L' \cdot \widetilde{\mathbb{J}} \\ {\scriptstyle s'}\big\downarrow & & \big\downarrow{\scriptstyle t'} \\ \mathbb{A} & & \mathbb{A}^\circ \times \mathbb{B} \end{array}$$

where

$L' = \big\{(L_0, \ell_0) \in \wp(L) \times L \mid L_0 \cap \{\ell_0\} = \emptyset, L_0 \cup \{\ell_0\} = L\big\}$;

$\widetilde{\mathbb{J}} = \oint \hom_{\mathbb{J}}$ (the twisted arrow category of $\mathbb{J}$);

$s' = [s \circ (\iota_0 \cdot \pi_2)]_{(L_0,\ell_0)\in L'}$ for $\iota_0 : L_0 \hookrightarrow L$; and

$t' = \big[\langle (s\, \iota_{\ell_0})^\circ \pi_1, t\, \pi_2 \rangle\big]_{(L_0,\ell_0)\in L'}$.